Towards human-centric software testing

Samantha Catania
Department of Computer Science
University of Malta
samantha.catania.12@um.edu.mt

Chris Porter
Department of Computer Information Systems
University of Malta
chris.porter@um.edu.mt

Mark Micallef
Department of Computer Science
University of Malta
mark.micallef@um.edu.mt

***Abstract*—Software testing is widely perceived to be the main activity in the software development process that provides confidence in the quality of a product prior to release. However, the term *software testing* itself provokes a multitude of different definitions and opinions as to the nature of the profession, the role of software testers and the utility of different processes and tools that come with the territory [1][2]. We argue that in order for researchers to effectively study the field and contribute to its progress, a consensus first needs to be reached about the entity being studied. In this paper we present an empirical study based on the modified Delphi card sort method involving four cohorts of testers in Malta and London. The result of this study is a consolidated consensus-based mental model outlining how software testers perceive their profession. This mental model can be used to align any future research efforts and tool development with testers' own perception of their context.**

***Index Terms*—Software testing, human factors and ergonomics, mental models**

I. Introduction

"Ergonomics [or human factors] is the scientific discipline concerned with the understanding of interactions among humans and other elements of a system, and the profession that applies theory, principles, data and methods to design in order to optimise human well-being and overall system performance." [3]

Testing is a human-intensive activity, requiring constantly changing information, in different formats, from various sources with different levels of access and over a variety of channels and tools. The level of experience and training background adds up to the complex nature of the domain, all of which contribute to high levels of cognitive workload. To date, and to the best of our knowledge, testing as a discipline in its various forms and shapes has never been treated scientifically from an ergonomics and human factors perspective. The terms 'ergonomics' and 'human factors' are interchangeable, however their use implies either the immediate physical environment (e.g. the workbench), or to the wider context in which work is carried out (e.g. the process), respectively [4].

In the past, we have attempted to treat this domain from both an ergonomics perspective (e.g. augmented reality in workbenches, just-in-time information and cognitive workload as well as from a human-factors perspective (e.g. the impact of training on tester performance [5], human-centred test frameworks). Throughout this period we also immersed ourselves in practice within industry teams, as much as possible, to be able to experience and discuss issues with industry professionals.

It transpires that there is a more pressing need that needs to be tackled first before we are able to further this line of investigation. Due to the complexity of the domain, different people will have different and often wide-ranging perceptions and expectations of testing in practice and theory. Therefore, to be able to reason in terms of human factors and ergonomics in testing, we first need to build a clear picture of how testers reason about testing in general. This paper aims to achieve this. We therefore present a mental-model built following an empirical exercise with software testing professionals in Malta and London using the modified Delphi card sorting method. This mental-model aims at providing a common vocabulary through which we can study problems and propose solutions which would ultimately benefit software testers and their well-being.

A mental model or a conceptual model, represents the "cognitive shorthand" [6] of how a person, or a group of people, understand a complex product or domain. Looking at this from the perspective of a website, this conceptual model is enough to help users navigate and interact with the site, however if the implementation varies widely from the users' mental model, users will find it close to impossible to reach their goals without having to think hard about their actions and how the site is reacting.

Although we understand that testing is complex in nature, we present a mental model based on a consensus-driven empirical exercise which elicits testers' beliefs about the domain and its various aspects. This aims to inform human factors and ergonomics researchers and designers in their effort to contribute towards this domain. Through an understanding of how testing professionals perceive this domain, contributions could be better aligned and communicated to resonate with the nature of their work. A designer's primary goal is to build user interfaces that map as closely as possible with the users' mental models, abstracting away the internal complexities [7]. This same principle applies for research efforts, whereby we first need to understand the primary stakeholders' beliefs and perceptions driving everyday work before being able to improve working conditions. Closing the gap between testers' mental models and researchers' perceptions or beliefs about the domain increases the chances of (a) improving the tester's

DOI reference number: 10.18293/SEKE2019-065

well-being by solving the right problems at the right level, and (b) adoption of research outcomes in industry. As in design, research efforts need to be aligned with the user's mental model. For this to happen, researchers need to use their expertise in research and frame their efforts as closely to the testers' mental model as possible, building a model which bridges their knowledge and efforts with testers' perceptions. Cooper defines this as the "represented model" which is "one of the designer's most important goals". Cooper states that "it is critical that designers understand in detail how their target users think about the work they do with the software" [6].

II. Software Testing

Software testing is an activity predominantly associated with ensuring that a product meets a certain level of quality before it is released to customers. Despite its widespread use and numerous practitioners worldwide, software testing is arguably the least understood part of the development process [2]. In the two decades that we have studied and practised in the field, the lack of consensus on the nature of the field, its processes, tools and practitioners' role has been an ever palpable characteristic. Many conversations that we have had about software testing decisions in the industry have involved phrases like *"I do not believe in automated testing"*, *"that is not what I think your role as a tester should be"* or *"we need to release tomorrow, can you test this quickly please?"*.

Even definitions by respected authorities differ, albeit with some overlap. The British Standards Index refers to software testing as the activity of exercising software with the intent of finding errors and verifying that is satisfies specified requirements [8]. The IEEE has a similar definition but includes the notion that a system can be tested without being exercised (inspection). The ISTQB syllabus [9] (the de facto standard for tester certification) makes reference to testing being a measurement of software quality.

Within the field itself, there are a multiplicity of roles, taxonomies, strategies, techniques, processes and tools. The correct time, place and usage of each of these elements is the subject of frequent debate, which has also given rise to different schools of thought within the industry. Engineering-driven schools treat testing as a standards-driven industry which practitioners can consequently be trained and certified in. On the other end of the scale, the "Context-Driven" school of thought claims that the value of any practice depends on its context and the concept of a one-size-fits-all definition of the field and its practices is misguided [10].

In a widely cited paper charting the progress and future direction of the field, Bertolino [1] makes reference to the multiplicity of meanings that arise from the term "software testing", as well as the particular research challenges that this fact generates. Bertolino goes on to set out the field's achievements to date, ongoing challenges and future dreams. The first dream she outlines is that of the development of a unified theory of testing.

This lack of clarity is a contributing motivation behind this work. Before one can design research, tools and processes that cater for the human tester more effectively, one first needs to understand the role and subsequent needs of the human tester. To this end, we undertook a consensus-based investigative approach as discussed in the following section.

III. Methodology

Motivated by the objective of establishing a consensus-based mental model amongst practitioners in the field about the nature of software testing, we selected a methodology centred around the Modified Delphi Card Sort method. The Delphi technique is a widely used and accepted method for gathering data from respondents within their domain of expertise [11]. Originally developed in the 1950s by Dalkey and Hemler [12], it has been widely used and adapted in various disciplines as a means of seeking out information that can generate consensus amongst participants.

Whereas initial versions of the technique involved using questionnaires with participants, we have chosen to use a card-sort variant whereby participants are asked to group concepts from their domain under categories using index cards. This can be done in one of two ways. The first is to utilise an open card-sort approach, whereby participants start with an empty slate and build a representation from there. The second way involves providing the first participant with a so-called *seeded deck*, that is, an initial model of the domain which she may choose to agree or disagree with in whole or in part. The participant can make any changes she sees fit by changing categories, adding terms and removing others. The result of the first participant's card sort are used as the seeded deck for the next participant, and so on. This technique has been shown to converge to a consensus with as little as 8-10 participants [13].

As depicted in Figure 1, two card sort exercises were carried out: one during a meeting of professional testers in Malta and another at a similar meeting organised by the British Computer Society in London.

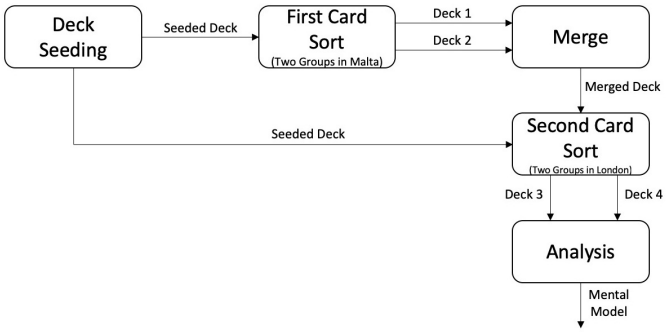

Fig. 1. An overview of the process used during the study.

Following an initial deck seeding exercise (Section III-A), two parallel card sort exercises were carried out (Section III-B) by two cohorts of participants, which resulted in two sorted decks that were subsequently merged (Section III-C). A second card sort (Section III-D) then took place, again with two parallel cohorts, one using the original seeded deck

and the other using the merged deck from the first card sort. This resulted in two final decks of cards which were fed into an analysis process (Section III-E) in order to produce the consolidated mental model. Each of these stages are discussed in turn below.

A. Deck Seeding

The seeded deck was compiled from a subset of terms found in a glossary published by the International Software Testing Qualifications Board (ISTQB) [14]. Given that the glossary contains over 600 terms, our initial processing of the glossary involved the removal of outlier terms and synonyms. Whilst outlier terms consisted of terms which we felt would not be helpful as part of the seeded deck (e.g. *Agile Manifesto* and *Myers-Briggs Type Indicator*), it is worth noting that should we be mistaken, it was entirely permissible for the terms to be reintroduced by participants during the card sort. At this point, we carried out a manual categorisation of the remaining terms into five broad categories: (1) *Testware* refers to entities that exist in the most part to enable and/or support the planning and execution of testing activities on a project; (2) *Artefact* refers to entities that are created, used and manipulated by testers or other stakeholders at some point during the testing process; (3) *Runtime:* refers to terms which represent entities that come into play in the period that a system is being executed as part of a test; (4)*Tools:* refers to devices, typically embodied as software systems that help support testers in a variety of tasks; and finally (5) *Test Strategy*, a category containing terms that identify different types of plans of action that testers can use to achieve their goal during software testing.

Once these categories were identified, we manually reduced the amounts of terms in each category based on our knowledge of the testing domain. Therefore, whereas a category called *Test Strategy* initially contained eighty terms, we manually reduced this to seven by keeping the most common strategies (e.g. *Acceptance Testing*) and removing the more obscure ones (e.g. *Monkey Testing*).

The resulting seeded deck consisted of five categories and twenty-five terms as shown in Table I.

TABLE I
THE CONTENTS OF THE INITIAL SEEDED DECK.

Category	Seeded Terms
Testware	Test Outcome, Test Data, Reports, Documentation, Test Suite, Test Script
Artefact	Features, Bugs, Code
Runtime	Configuration, System Under Test, Environment, Tester
Tools	Static Analysis Tools, Coverage Tools, Performance Testing Tools, Bug Tracking Tools, Automated Testing Tools
Test Strategy	Acceptance Testing, Ad Hoc Testing, Branch Testing, Exploratory Testing, Integration Testing, Performance Testing, Regression Testing

The materialisation of this seeded deck enabled us to move on to the next stage of the study.

B. First Card Sort - Malta

Armed with two copies of the seeded deck, we attended a gathering of software testing professionals in Malta and recruited sixteen participants, who we split evenly into two cohorts. Each participant was asked to reflect on their mental model of software testing and subsequently spend as much time as needed to modify the deck so that it accurately fit his/her mental model. This was done by any combination of (1) *adding* new terms or groups; (2) *moving* cards between groups; and (3) *removing* terms or groups which they considered not to be a fit for purpose. When a participant was finished, the next participant was asked to come in and continue the exercise.

At the end of the exercise, the two decks were analysed and merged into a single deck as discussed in the following section.

C. Merging of Decks

The two decks resulting from the parallel card sorts carried out in Malta were merged using a five-step process as follows. The first step involved *discarding replicated terms* such that terms which were added multiple times in the same deck were earmarked for discarding so that only one instance of each term remained. In cases where the terms appeared in different categories, reference to the ISTQB Glossary [14] was made in an effort to choose the category that best fit the emerging mental model. The next step involved *renaming of synonyms* to make sure that any instances whereby testers were referring to the same concept using different labels, one term was selected and used in place of all the synonyms. We then turned our attention to categories and automatically *retained any categories that were common in both decks* for the merged deck. Similarly we also *retained terms that appeared in both decks*. If the terms appeared in different groups, a judgement was made as to which group was the best fit for that term. Finally, we *discarded any empty categories*. Any categories that contained no terms following the preceding steps in the merge were discarded.

The two card sort exercises in Malta both retained the original five categories from the seeded deck and added four groups between them. However, the merging process resulted in the four new categories having no terms and were consequently removed. This resulted a merged deck with the characteristics described in Table II.

TABLE II
THE NUMBER OF INITIAL (I), RETAINED (R), ADDED (A), DELETED (D) AND FINAL (F) TERMS FOLLOWING THE FIRST CARD SORT AND MERGE.

Category	I	R	A	D	F
Testware	6	5 (83%)	2 (33%)	1 (17%)	7 (+17%)
Artefact	3	2 (67%)	5 (166%)	1 (33%)	7 (+133%)
Runtime	4	3 (75%)	2 (50%)	1 (25%)	5 (+25%)
Tools	5	4 (80%)	8 (160%)	1 (20%)	12 (+140%)
Test Strategy	7	5 (71%)	7 (100%)	2 (29%)	12 (+71%)
Totals:	**25**	**19 (76%)**	**24 (96%)**	**6 (24%)**	**43 (+72%)**

The figures in Table II indicate that there was an overall increase of 72% in the size of the mental model with about

24% of our initial seeded deck being rejected by participants. The highest relative increases where in the *Tools* category (+140%) and the *Artefact* category (+133%).

D. Second Card Sort - London

In order to further refine the emerging mental model, two cohorts of seven volunteers attending a one-day software testing conference organised by the British Computer Society in London were tasked with carrying out a further card sort. One cohort started with the original seeded deck whilst the other one started with the merged deck. The exact same protocol that was used in the first card sort was utilised in the second card sort with the results being fed into the final stage of the study.

E. Analysis

The card sort exercises generated 83 terms in total and each was either seeded, added, removed or moved at one or more points during this study. In order to make sense of this data, a scoring system was devised whereby every time participants interacted with a term, the term would gain or loose a certain amount of points. Therefore, when a term was seeded by a researcher, added by a participant or removed by a participant, it gained 1 point, 2 points and lost one point respectively. Whilst these values may seem arbitrary, they were designed to (1) characterise each term's journey through the study into a single score; and (2) assign more importance to terms which were added by participants without any prior mention.

The result was a ranked list of 83 terms in five categories which was then analysed by researchers with a view of establishing a cutoff point in each category that would indicate which terms should be included in the consolidated mental model. The cutoff decision was made based on (1) identifying points in the list where a significant scoring gap occurred between one term and the next; and (2) manually removing terms which objectively did not fit the category or were synonyms for other terms that were already included. The result was a mental model with the same categories as those designed in the original seeded deck.

IV. Discussion

The mental model resulting from the research is respresented in the form of a mind map in Figure 2. In this section, we discuss characteristics of this mental model and comment on its convergance, evolution and its implications on testers' perception of their field.

A. Convergence

Consistent with other studies that used the Modified Delphi Card Sort method, participants appear to converge towards a consensus relatively quickly. As shown in Table III, whereas the first card sort resulted in an increase in deck size from 25 to 43 terms (72%), there was only a net decrease of 2 terms (-5%) in deck size following the second card sort. Also, as shown in Table IV, more terms were retained, and less were added or removed as research progressed from the first to the second card sort.

TABLE III
COMPARING MENTAL MODEL SIZE FROM THE SEEDED DECK (Δ_s) THROUGH TO THE MERGED DECK (Δ_m) AND THE FINAL MENTAL MODEL.

	Seeded	**Merged**		**Final**		
Category	#	#	Δ_s	#	Δ_s	Δ_m
Testware	6	7	+17%	5	-17%	-29%
Artefact	3	7	+133%	10	+233%	+43%
Runtime	4	5	+25%	6	+50%	+20%
Tools	5	12	+140%	9	+80%	-25%
Test Strategy	7	12	+71%	11	+57%	-8%
Totals:	25	43	+72%	41	+64%	-5%

TABLE IV
TERMS RETAINED, ADDED AND DELETED BETWEEN SUCCESSIVE DECKS.

	Seeded → Merged	**Merged → Final**
Retained	19 (76%)	37 (86%)
Added	24 (+96%)	4 (+9%)
Deleted	6 (-24%)	6 (-14%)
Net Churn:	18 (+72%)	-2 (-5%)

B. How the seeded model evolved

The contents of the Testware category remained mostly unchanged with no terms being added and one term (*Documentation*) being moved to the Artefacts category. The Artefacts category itself grew threefold in size from three terms to nine with practitioners strongly backing terms that the original seeded model had left out. Interestingly, the term *Features* was consistently removed by participants but was replaced by multiple other terms such as *Specification* and *Acceptance Criteria*.

The Runtime category retained all but one of its seeded terms (*Tester*) and doubled in size from three to six terms with the testers introducing the terms *Mocking*, *User* and *Bugs*.

Finally, practitioners retained most of the terms from the Tools and Test Strategies categories, removing two terms from each category but also supplemented each category considerably. Six new types of tools were added, as well as six new testing strategies. All terms added received consistent backing from participants in successive card sorts.

C. Interesting Observations

The process of obtaining information about the mental model of software testers from practitioners themselves provided some interesting insights.

Firstly, testers do not see themselves as part of their own mental model. The term *Tester* was presented to three cohorts as a result of being part of the seeded deck. However, it was removed every single time. This indicates to us that contrary to our view that the tester should be at the centre of any discussion related to the testing process, the testers take themselves out of the equation. This flies contrary to many discussions we witnessed in the industry whereby testers would express frustration at not being included enough in the software development process.

Secondly, testers need to somehow effectively use as many as eleven different testing strategies in their day-to-day job. This is complemented by nine different types of testing tools,

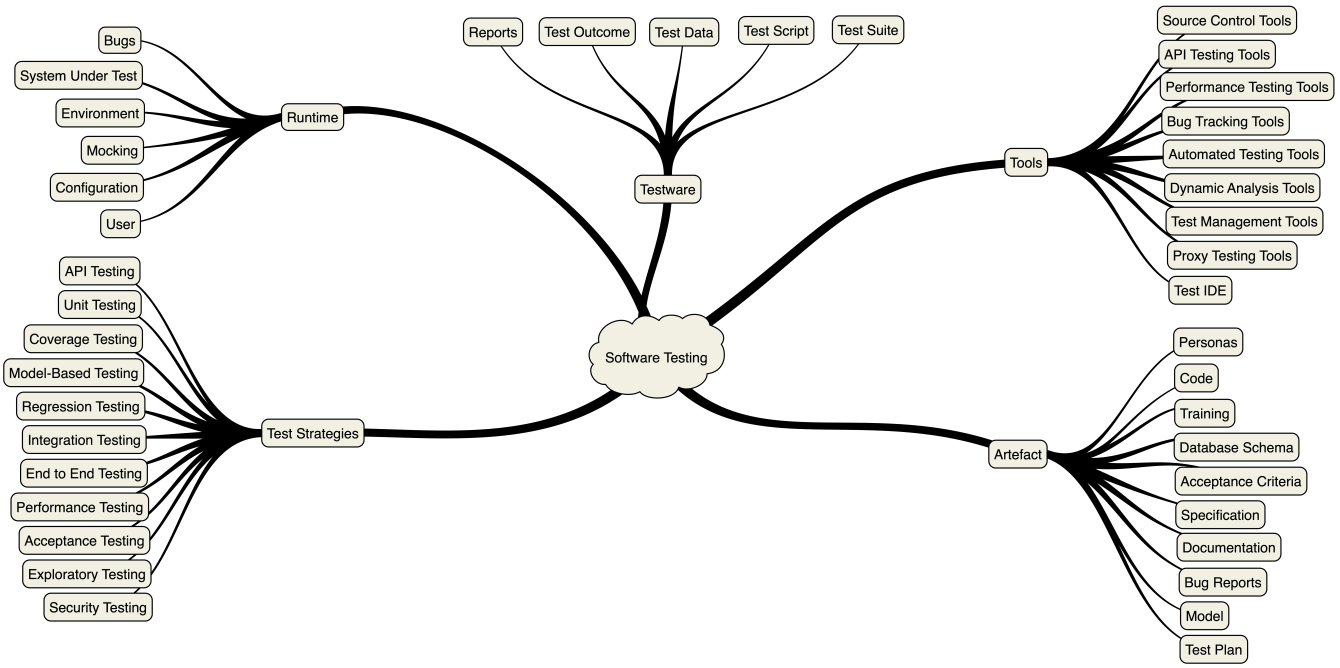

Fig. 2. The tester's mental model that resulted from the card sorts.

which hopefully make the testers' lives easier. This indicative of the highly complex nature of the testing profession.

In the multiple opportunities that presented themselves throughout this study (from seeding through multiple card sorts), the term *Usability Testing* was only mentioned once and did not make it into the consolidated mental model. This is interesting because it indicates that testers do not perceive usability testing as a core part of their job. Rather, they are concerned with ensuring that the product meets the stated specifications, even if the specifications do not necessarily provide the customer with the experience they desire. Given the ease with which customers can move to competitors in today's dynamic online markets, this type of thinking could be counterproductive and end up producing high quality software that customers do not want [15].

V. Related Work

In this section, we compare and contrast our work with similar research efforts. Different works in the literature concerned with understanding the software testing domain exist for one of three reasons: (1) understanding the application of testing in a specific domain; (2) knowledge management; or (3) forming a more coherent understanding for pedagogical purposes.

With regards to domain-specific work, Nasser et al. [16] present an ontology based on state machine based testing whilst Sapna and Mohanty's [17] work focuses on scenario-based testing. Yu et al. [18] focus on understanding software testing as a service (TAAS). Whilst such studies have value at forming an in-depth understanding of testing within individual domains, our work is more focused on forming a wider practitioner-oriented understanding of the field. Nevertheless, it is interesting to note how the organisation of different models differs based on their focus. For example, Yu et al. [18] proposed categories such as *Test Type*, *Target Under Test*, *Test Environment* and *Test Schedule*. Whilst these categories bare resemblance to those in our work and the work of others, the emphasis can be seen to focus on scenarios whereby testing is perceived (and even sold) as an outsourced service.

Focusing on increasing testcase reuse through knowledge management, Guo et al. [19] develop an ontology centred soley around the *test case*. They develop and propose the use of their unified standard format for test cases and argue that this can promote reuse. Focusing on web services, Bai et al. [20] propose a so-called Test Ontology Model that models testing artefacts and relationships between them. Barbosa et al. [21] propose *OntoTest* a collection of six sub-ontologies of testing named as *Testing Process*, *Testing Phase*, *Testing Artifact*, *Testing Step*, *Testing Resource*, and *Testing Procedure*.

Perhaps the work that is most closely related to this paper is that by Arnicans and Straujums [22] who propose a hierarchical model of the testing domain constructed from the 800 entries in the ISTQB Glossary [14], the same source used for creating the seeded deck in our study. Using a technique whereby each term in the glossary was assigned a weight and subsequently related to other words, they converted the ISTQB Glossary into a browsable hierarchical concept map. They found that the 'weightiest' nine terms where *testing*, *test*, *tool*, *software*, *process*, *analysis*, *capability*, *technique* and *coverage*. These top-level terms contain 425 (70%) of the glossary's entries between them. There are some similarities between Arnicans and Straujums' top terms and the categories in our mental model but the mapping is not direct. For example, the

term *tool* maps to our *tools* category whilst *software* maps to *artefact* and *technique* maps to *test strategies*. However, the terms contained in each of these mapped categories are not the same. One example is that some techniques that we refer to as strategies, are referred to as being part of the *process* category in Arnicans and Straujum's work. One should note that the scope behind the work differs from ours in that whilst Arniscans and Straujum are concerned with making the ISTQB Glossary more understandable, we are interested in understanding the mental model held by practitioners in the field.

VI. Conclusions and Future Work

We motivated this study by arguing that since testing is a human-intensive activity, the human tester needs to be at the centre of research contributions to the field. This implies that testing, as a discipline, would benefit from being treated scientifically from an ergonomics and human-factors perspective. However, our initial work in this area uncovered a lack of clarity as to what exactly constitutes software testing from the tester's perspective.

As a result of the study presented in this paper, we propose a mental model elicited from practitioners using a consensus building approach. Having this mental model available provides researchers with an insight into how testing practitioners perceive their professional context and can thus form as a basis for aligning research efforts with practitioners' views. The mental model by no means covers the whole testing domain but we argue that aligning research efforts to this model is more likely to provide value to the practitioners who created it.

A. Future Work

With regards to future work, we would like to pursue two main paths of research. Firstly, we would like to continue to validate the model through further card sorts in order to reduce external threats to validity whilst also gaining a deeper understanding about whether cohorts of testers with specific characteristics (e.g. experience testers or testers working in a specific domain) would diverge from the model. Our initial discussions with peers about the results presented here resulted in questions regarding issues such as how the background of participants might have affected the results or why certain terms do not appear. It would be interesting to investigate these questions and also repeat these exercises on a regular basis to understand if and how practitioner perceptions evolve. Secondly, we would like to revisit our work on treating software testing from an ergonomics and human factors perspective (e.g. augmented reality workbenches, just-in-time information and cognitive workload) in light of the mental model and its implications. This will help us refocus such efforts so as to increases the chances of (a) improving the tester's well-being by solving the right problems at the right level, and (b) adoption of research outcomes by industry stakeholders.

References

[1] A. Bertolino, "Software testing research: Achievements, challenges, dreams," in *2007 Future of Software Engineering*. IEEE Computer Society, 2007, pp. 85–103.

[2] J. A. Whittaker, "What is software testing? and why is it so hard?" *IEEE software*, vol. 17, no. 1, pp. 70–79, 2000.

[3] I. E. Association, "Definition and domains of ergonomics," 2016. [Online]. Available: https://www.iea.cc/whats/

[4] C. I. of Ergonomics and H. Factors, "What is ergonomics? find out how it makes life better." [Online]. Available: https://bit.ly/2SoLsJz

[5] M. Micallef, C. Porter, and A. Borg, "Do exploratory testers need formal training? an investigation using hci techniques," in *2016 IEEE Ninth International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*. IEEE, 2016, pp. 305–314.

[6] A. Cooper, R. Reimann, and D. Cronin, *About face 3: the essentials of interaction design*. John Wiley & Sons, 2007.

[7] J. Nielsen, "Mental models," 10 2010. [Online]. Available: https://www.nngroup.com/articles/mental-models

[8] S. C. Reid, "Bs 7925-2: The software component testing standard," in *apaqs*. BSI, 2000, p. 139.

[9] A. Roman, "2018 foundation syllabus overview," in *A Study Guide to the ISTQB® Foundation Level 2018 Syllabus*. Springer, 2018, pp. 3–11.

[10] C. Kaner and J. Bach, "What is context-driven testing," 2009.

[11] C.-C. Hsu and B. A. Sandford, "The delphi technique: making sense of consensus," *Practical assessment, research & evaluation*, vol. 12, no. 10, pp. 1–8, 2007.

[12] N. Dalkey and O. Helmer, "An experimental application of the delphi method to the use of experts," *Management science*, vol. 9, no. 3, pp. 458–467, 1963.

[13] A. Soranzo and D. Cooksey, "Testing taxonomies: beyond card sorting," *Bulletin of the Association for Information Science and Technology*, vol. 41, no. 5, pp. 34–39, 2015.

[14] I. ISTQB, "Glossary of testing terms," *ISTQB Glossary http://www. istqb. org/downloads/finish/20/193. html*, 2015.

[15] E. Ries, *The lean startup: How today's entrepreneurs use continuous innovation to create radically successful businesses*. Crown Books, 2011.

[16] V. H. Nasser, W. Du, and D. MacIsaac, "Knowledge-based software test generation." in *SEKE*, 2009, pp. 312–317.

[17] P. Sapna and H. Mohanty, "An ontology based approach for test scenario management," in *International Conference on Information Intelligence, Systems, Technology and Management*. Springer, 2011, pp. 91–100.

[18] L. Yu, L. Zhang, H. Xiang, Y. Su, W. Zhao, and J. Zhu, "A framework of testing as a service," in *2009 International Conference on Management and Service Science*. IEEE, 2009, pp. 1–4.

[19] S. Guo, J. Zhang, W. Tong, and Z. Liu, "An application of ontology to test case reuse," in *2011 International Conference on Mechatronic Science, Electric Engineering and Computer (MEC)*. IEEE, 2011, pp. 775–778.

[20] X. Bai, S. Lee, W.-T. Tsai, and Y. Chen, "Ontology-based test modeling and partition testing of web services," in *2008 IEEE International Conference on Web Services*. IEEE, 2008, pp. 465–472.

[21] E. F. Barbosa, E. Y. Nakagawa, and J. C. Maldonado, "Towards the establishment of an ontology of software testing," in *SEKE*, 2006, pp. 522–525.

[22] G. Arnicans and U. Straujums, "Transformation of the software testing glossary into a browsable concept map," in *Innovations and Advances in Computing, Informatics, Systems Sciences, Networking and Engineering*. Springer, 2015, pp. 349–356.